Poster III-10

A Query Language To Support Scientific Discovery

Eckman, Barbara[1], Deutsch, Kerry[2], Gaasterland, Theresa[3], Janer, Marta[2], Lacroix, Zoé[*4], Raschid, Louiqa[5]

[1]IBM Life Sciences, West Chester, PA, USA; [2]Institute for Systems Biology, Seattle, WA, USA; [3]Rockefeller University, New York, NY, USA; [4]Arizona State University, Tempe, AZ, USA; [5]University of Maryland, College Park, MD, USA

Data management is of critical importance in bioinformatics. If the central task of bioinformatics is the computational analysis of biological sequences, structures, and relationships, it is crucial that biological sequences and all associated data be accurately captured, annotated, and maintained, even in the face of rapid growth and frequent updates. It is also critical to be able to retrieve data of interest from multiple distributed heterogeneous data sources in a timely manner, and precisely enough to be able effectively to separate them from the distracting noise of irrelevant, unreliable or insignificant data.

Traditional database approaches have limitations in their ability to support scientific discovery [1]. A query language that supports biological science must be capable of expressing and implementing biological investigations. Most scientific discoveries are supported by a hypothetico-deductive reasoning [2,3] that is a modular process composed of a causal scientific question, a proposed explanation or hypothesis, an experiment aiming to test an hypothesis, the predicted results or expectations, the experimental results, and a conclusion. As scientific discovery has shifted from the traditional wet-lab where most of the tasks were performed by scientists to a reasoning exploiting digital data and computer technology, the access to digital data repositories and exploitation of digital data plays an ever increasing role and may benefit each module of scientific reasoning. However, the mapping between the scientific protocols expressed through complex workflows and the actual queries against databases is often difficult and involve several levels of translations from the scientist's needs to a query expressed in SQL or other query language.

A declarative language is in theory easier for biologists to use in implementing their own investigations, without relying on professional programmers. In practice, however, the types of operations that biological investigators typically need to perform often require rather complex SQL, and there is no easy way to map between the biological semantics and the SQL expressions necessary to capture them. For example, querying a table of gene expression results for genes relatively specific to pancreas (i.e., expressed in pancreas and only, say, 3 other tissues) requires a sub-query with GROUP BY and HAVING operators; the ubiquity of variant types in biological databases often requires outer joins and the use of functions like coalesce(A,B), which returns A if it is non-null, but B otherwise; and because greater confidence is placed in data attested by multiple sources, biologists often want to return only objects found in multiple selection sets defined by multiple sub-queries.

The Biological Query Language (BQL) aims to enhance the scientists querying ability by (1) providing an intermediate query language between scientific workflows and traditional query languages such as SQL, (2) expressing ranking and validating, operations not made directly available by traditional query languages and often difficult to express (by complex queries), and (3) constraining the evaluation of their operators by various semantics. It is composed of five operators: *collectio*n, *filterin*g, *rankin*g, *grouping* and *validatio*n. Intuitively, the *collection* operator consists in collecting data possibly from various data sources to increase the number of studied entries or to augment the information related to entries already collected. The *collection* operator may be expressed by relational joins. The *filtering* operator aims at reducing the number of entries or the information related to the entries. It may be expressed by a relational selection or a projection operator. The *ranking* operator consists in ordering collected entries with respect to a particular criterion. Traditional database systems offer various ranking mechanisms that the BQL ranking operator may be mapped to. In addition, many scientific queries involve applications such as textual search engines or BLAST that return ranked data. The BQL ranking operator exploits the available ranking methods and combines them where necessary. The *validation* operator consists in identifying among all collected entries the ones that are validated because they belong to two different paths and/or independent data sources. The fact that entries can be found in at least two different independent sources makes them more reliable and contributes to the scientific hypothetico-deductive reasoning.

Conclusion

Traditional data management approaches need to be leveraged to support scientific discovery. The Biological Query Language presented in this poster offers several enhancements including:

- scientist-friendly operators,
- supporting the expression of scientists' queries,
- enabling the use of various semantics exploiting the non-determinism of scientific resources,
- interfacing with traditional query languages, and
- providing a framework for optimization techniques as identified in [4].

References

1. Z. Lacroix and T. Critchlov (2003) *Bioinformatics: Managing Scientific Dat*a, Morgan Kaufmann
2. E. Lawson (1999) *The Living World - BIO 10*0, McGraw-Hill Primis
3. E. Lawson (2002) What does Galileo's Discovery of Jupiter Moons tell us about the Process of Scientific Discovery, *Science and Educatio*n, Volume 11, 1-24.
4. Eckman and Z. Lacroix and L. Raschid (2001) Optimized Seamless Integration of Biomolecular Data, in proc IEEE International Symposium on Bio-Informatics and Biomedical Engineering (BIBE), Washington, DC, 23-32.